1.1 Описание предметной области

Анализ дорожно-транспортных происшествий дает возможность отследить динамику изменений дорожных ситуаций в городе, выявлять особо опасные места их возникновения, а также причины и следствия, которые они несут за собой.

Дорожно-транспортные происшествия напрямую зависят от внешних факторов, таких как общая ситуация на дороге, тип участка дороги, погодные условия, часы суток, интенсивность потока машин и пешеходов.

Программная система «Car Crash Today» позволяет пользователю наблюдать за изменением ситуации на дороге и конкретных опасных участках за определённый период времени. Приложение берет данные о ДТП из сервиса «Яндекс.Карты» и обладает возможностью занесения данных пользователями системы с учетом модерации.

Целью программной системы «CarCrashToday» является агрегация новостей о ДТП как путем их добавления пользователями, так и в реальном времени из сторонних ресурсов, например, «Яндекс.Карты», а также предоставление статистики за выбранный период на дорожном объекте начиная от перекрестка или участка дороги до района или города в целом.

Функции ПС:

* представление подробных данных о ДТП в течении последних 3х лет, в любой отрезок времени;
* предоставление данных о конкретном ДТП из перечня;
* вывод информации о самых частых местах дорожно-транспортных происшествий;
* генерация графиков, отображающих динамику возникновения ДТП.

Диаграмма Гантта:

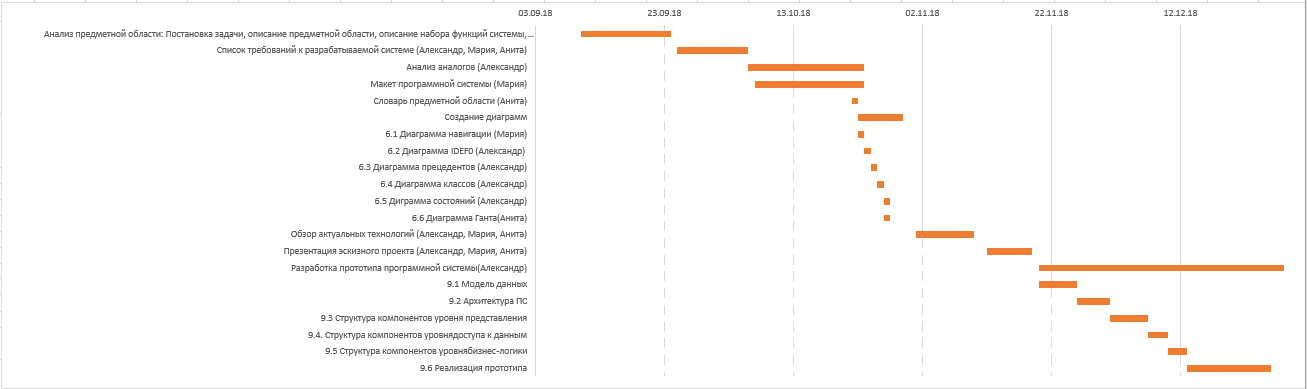


Рисунок 1. Диаграмма Гантта.

1.2 Основные понятия предметной области

Пользователь – лицо, зарегистрированное в системе.

Модератор – пользователь, с расширенными правами доступа, осуществляющий фильтрацию, проверку и вывод новостей.

Администратор – модератор, с расширенными правами доступа, осуществляющий управление системой и назначение ролей.

Дорожно-транспортное происшествие – запись, содержащая сведения о ДТП.

Анализ ДТП – вывод данных о произошедших ДТП за выбранный период времени на дорожном объекте начиная от перекрестка или участка дороги до района или города в целом.

Дорожный объект – место возникновения ДТП (участок дороги, перекресток).

1.3 Функциональные требования к системе

Необходимо было сформулировать конкретные функциональные требования к системе.

Система должна предоставлять пользователю обещанный функционал, а именно:

Бизнес требования:

* Веб-приложение для анализа ДТП.

Пользовательские требования:

* При первом посещении сайта пользователю должна быть доступна главная страница с информацией о приложении и различными функциями (которые становится доступны после регистрации и входа)
* Для авторизованного пользователя доступно: вывод анализа ДТП и графиков, добавление новостей о ДТП.

После определения конкретных функциональных требований можно переходить к их анализу.

1.4 Нефункциональные требования к системе

Сформулируем нефункциональные требования, в которых определимся с выбором языка программирования, платформы, структуры приложения и других деталей предстоящей разработки.

Ограничения:

* Разработка системы должна быть на языке Java.
* Система должна представлять из себя веб-приложение.
* Система должна использовать технологию Spring.
* Для хранения данных используется СУБД MySQL.
* Пользовательский интерфейс должен быть удобным, чтобы каждый пользователь мог интуитивно понять на какую кнопку нажать, чтобы совершить нужное действие.
* Система должна быть надёжной, т. е. должна сохранять работоспособное состояние на протяжении длительного времени, иметь небольшое время реакции. Она должна выдерживать не менее 120 пользователей в минуту

1.5 Сравнительный анализ аналогов

Название: Стат ГИБДД

Интерфейс представлен на рисунке 2.

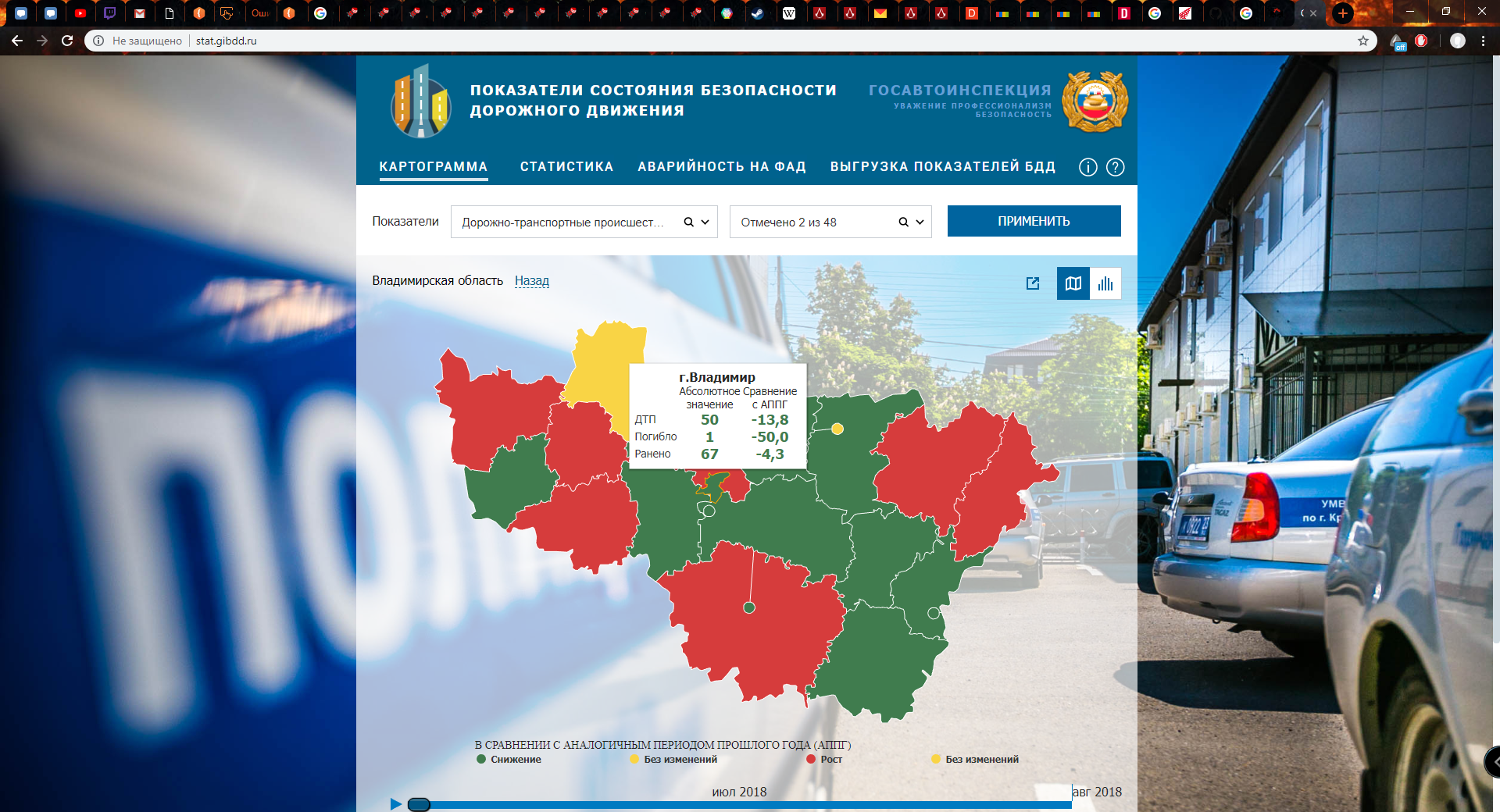


Рисунок 2. Интерфейс Стат ГИБДД.

Преимущества. Большая база данных о ДТП, непосредственный источник информации.

Недостатки. Обобщенные сведения о ДТП, отсутствие детального анализа происшествия, ошибки в данных.

Таблица 1. Сравнение аналогов

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Название** | **Оценка интерфейса** | **Оценка юзабилити** | **Скорость обновления информации** | **Модерация информации** |
| Стат ГИБДД | 5 из 5 (качественная визуализация) | 4 из 5 (сложная навигация) | 3,5 из 5 (информация о дтп появляется спустя некоторое время (от 3 дней)) | 3,5 из 5 (информация содержит орфографические ошибки и неточные цифровые данные) |

1.6 Обзор технологий

Основной целью фреймворка Spring Framework является упрощение разработки корпоративных приложений на языке Java за счет использования приемов внедрения зависимостей, аспектно-ориентированного программирования и устранения необходимости писать шаблонный программный код. Даже если бы возможности фреймворка Spring ограничивались этим набором, его уже стоило бы использовать.

Доступ к данным и интеграция:

Работа с JDBC зачастую сводится к обширному использованию шаблонного кода, который устанавливает соединение, создает SQL- запрос, обрабатывает результаты запроса, а затем закрывает соединение. Модуль поддержки JDBC объектов доступа к данным (Data Access Objects, DAO) в Spring абстрагирует шаблонный код и позволяет сохранить простым и прозрачным программный код, реализующий операции с базами данных, а также предотвратить проблемы, возникающие в результате ошибки освобождения ресурсов. Этот модуль также образует слой важных исключений, основанных на сообщениях об ошибках, посылаемых некоторыми серверами баз данных. Благодаря этому вам больше не придется расшифровывать непонятные сообщения об ошибках SQL!

Для тех, кто предпочитает использовать инструменты объектно-реляционного отображения (Object-Relational Mapping, ORM) поверх JDBC, фреймворк Spring предоставляет модуль ORM. Поддержка ORM в Spring основана на поддержке DAO, обеспечивающей удобный способ создания объектов доступа к данным для некоторых ORM-решений. Фреймворк Spring не пытается реализовать свое собственное ORM-решение, а просто предоставляет рычаги управления некоторыми популярными фреймворками ORM, включая Hibernate, Java Persistence API, Java Data Objects и iBATIS SQL.

Веб и удаленные взаимодействия:

Парадигма модель–представление–контроллер (Model–View– Controller, MVC) часто используется при создании веб-приложений, в которых пользовательский интерфейс отделен от логики работы приложения. Язык Java не испытывает недостатка в фреймворках MVC, благодаря поддержке Apache Struts, JSF, WebWork и Tapestry, являющихся наиболее популярными реализациями MVC.

1.7 Функции системы

Анализ ДТП

На рисунке 3 представлена диаграмма прецедентов.



Рисунок 3. Диаграмма прецедентов.

**1.7.1 Авторизация пользователя в системе**

Название: «Войти»

Действующее лицо: «Гость»

Предусловие: Гость должен быть зарегистрирован в системе.

Основной поток:

Пользователь открывает главную страницу нажимает кнопку «Войти», и переходит на страницу отображающую форму входа на сайт;

Пользователь заполняет поле логин и поле пароль и авторизуется в системе.

Альтернативный поток: данные заполнены некорректно.

Гость ввёл данные некорректного формата или заполнил не все обязательные поля;

Вывод сообщения об ошибке.

Постусловие: Гость авторизован в системе.

**1.7.2 Регистрация пользователя в системе**

Название: «Регистрация»

Действующее лицо: «Гость»

Основной поток:

1. Гость открывает страницу, отображающую форму регистрации;

2. Гость заполняет все необходимые данные и регистрируется в системе.

Альтернативный поток: данные заполнены некорректно.

1. Гость ввёл данные некорректного формата или заполнил не все обязательные поля;

2. Вывод сообщения об ошибке.

Постусловие: Пользователь зарегистрирован в системе.

**1.7.3 Просмотр новостей**

Название: «Новости»

Действующее лицо: «Пользователь»

Основной поток:

Пользователь нажимает на кнопку «Новости», открывается страница с последними новостями о ДТП.

Альтернативный поток: -.

Постусловие: Пользователь получил данные.

1.8 Анализ бизнес-процесса «»

Для большего понимания рассмотрим нашу систему в нотации IDEF0 [8]. Приведем два уровня декомпозиции.

Перейдем к диаграмме А-0 бизнес-процесса «Прогноз» (рисунок 4).



Рисунок 4. Диаграмма A-0 бизнес процесса «Анализ ДТП» в нотации «TO-BE».

Опишем бизнес-процесс «Анализ»:

Вход – материал или информация, которые станут результатом на выходе. На данной диаграмме это «Данные о ДТП».

Управление – нормативные, регламентирующие и управляющие данные, которыми руководствуется процесс во время его выполнения. На данной диаграмме это «Правила пользования приложения».

Механизм – ресурсы, необходимые для выполнения процесса. На данной диаграмме это «Приложение Car Crash Today», «Пользователь».

Выход – информация или материал, которые являются результатом работы. На данной диаграмме это «Готовый анализ».

Рассмотрим декомпозицию этого процесса (рисунок 5).



Рисунок 5. Декомпозиция процесса «Анализа ДТП» в нотации «TO-BE».

Как видно из диаграммы, процесс разбился на три подпроцесса.

Чтобы подробнее разобрать эти подпроцессы, проведем декомпозицию каждого. Рассмотрим процесс «Выбор периода прогноза» (рисунок 6).



Рисунок 6. Диаграмма A-1 декомпозиция процесса «Выбор периода прогноза» в нотации «TO-BE».

Рассмотрим декомпозицию процесса «Получение анализа» (рисунок 6).



Рисунок 7. Диаграмма A-2 декомпозиция процесса «Получение анализа» в нотации «TO-BE».

1.9 Диаграмма состояния

На диаграмме состояний (рисунок 8) отражается процесс формирования новости о ДТП.



Рисунок 8. Диаграмма состояний.

1.10 Общие принципы организации системы

В процессе анализа требований и предметной области архитектура системы была декомпозирована на несколько модулей, которые в свою очередь разделены на несколько подсистем. Каждый модуль и подсистема выполняют свою определенную функцию:

Модуль бизнес-логики – осуществляет основную логику системы:

* Подсистема взаимодействия с сервисом Яндекс.Карты – производит запись и извлечение данных из веб-сервиса;
* Подсистема доступа к данным – реализует необходимые CRUD-операции с классами-сущностями;

Веб-модуль – предоставляет конечному пользователю веб-интерфейс для работы с системой, обрабатывает HTTP запросы и ответы.

**1.11 Организация доступа к данным**

На рисунке 9 представлена диаграмма классов сущностей. На ней подробно видно, какие классы сущностей представлены в системе.



Рисунок 9. Диаграмма классов.

1.12 Взаимодействие компонентов системы

При выполнении сложных функций приложения обычно используется несколько компонентов. Для понимания как выполняется та или иная функция необходимо спроектировать и визуализировать взаимодействие компонентов системы.

Взаимодействие компонентов системы отражено с помощью диаграммы последовательностей (рисунок 10).



Рисунок 10. Диаграмма последовательностей.

1.13 Организация управления доступом

В системе предусмотрено три вида пользователей:

* пользователь;
* модератор;
* администратор.

У каждого пользователя есть свои строго определенные права, которые регулируют доступ к функциям системы. Например, пользователь может создать новость о ДТП, получить анализ о произошедших ДТП. Модератор может добавить новость на сайт и в БД. Администратор может редактировать пользователей.

**1.14 Навигация**



Рисунок 11. Схема навигации.

**1.15 Макет**

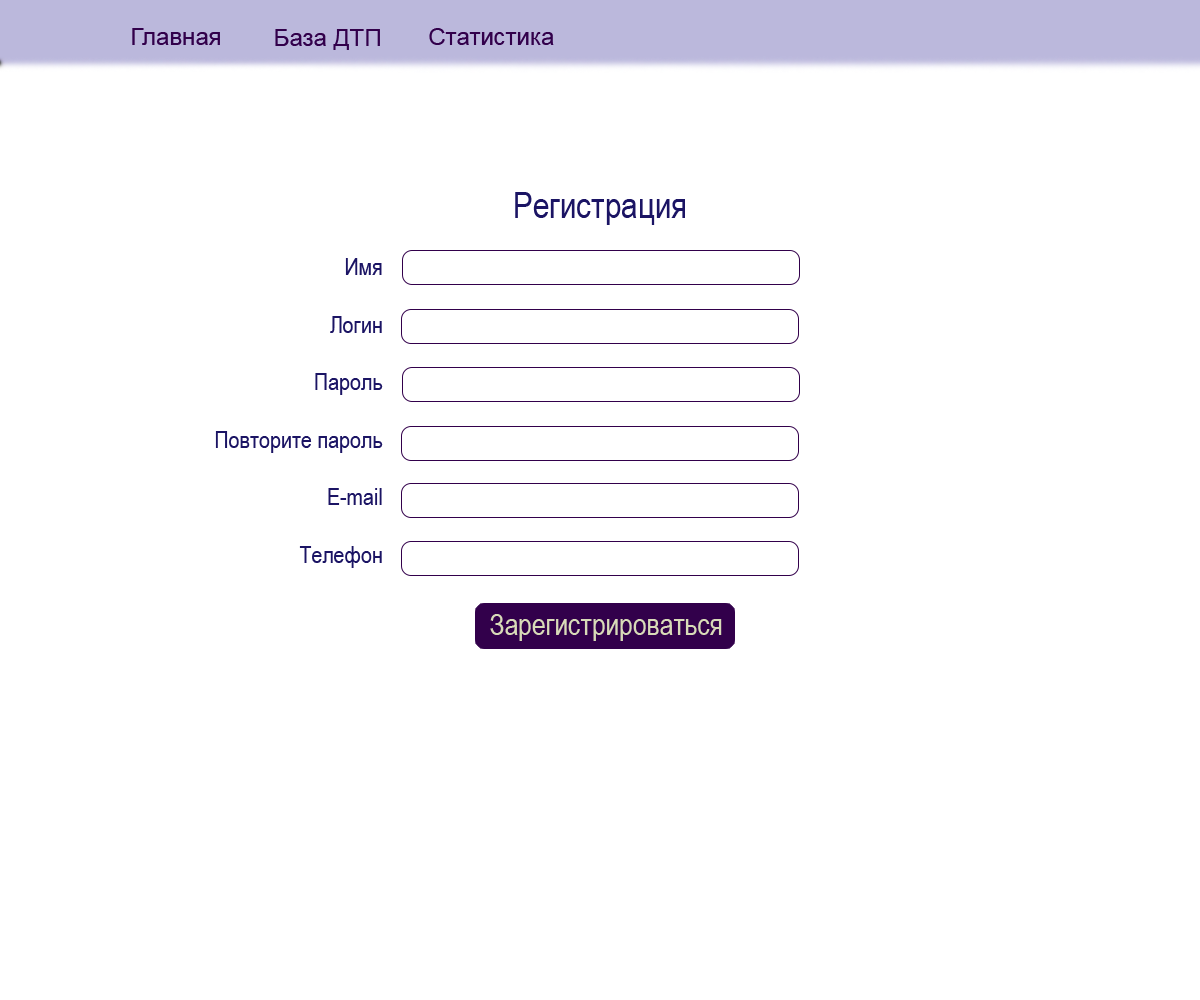


Рисунок 12. Форма регистрации.

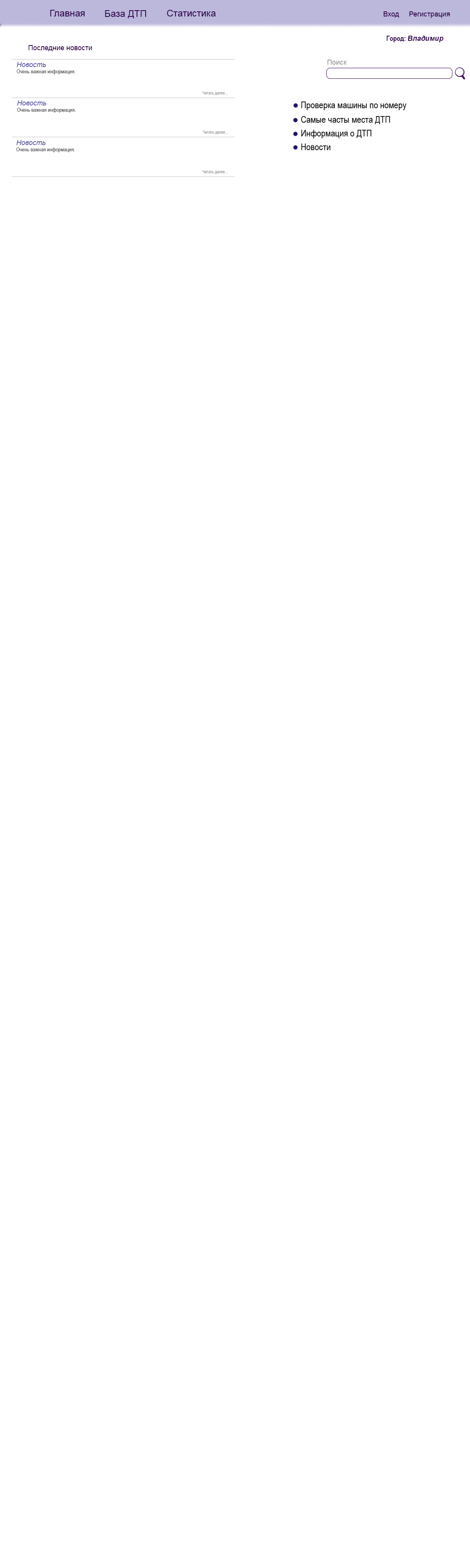


Рисунок 13. Главная страница.

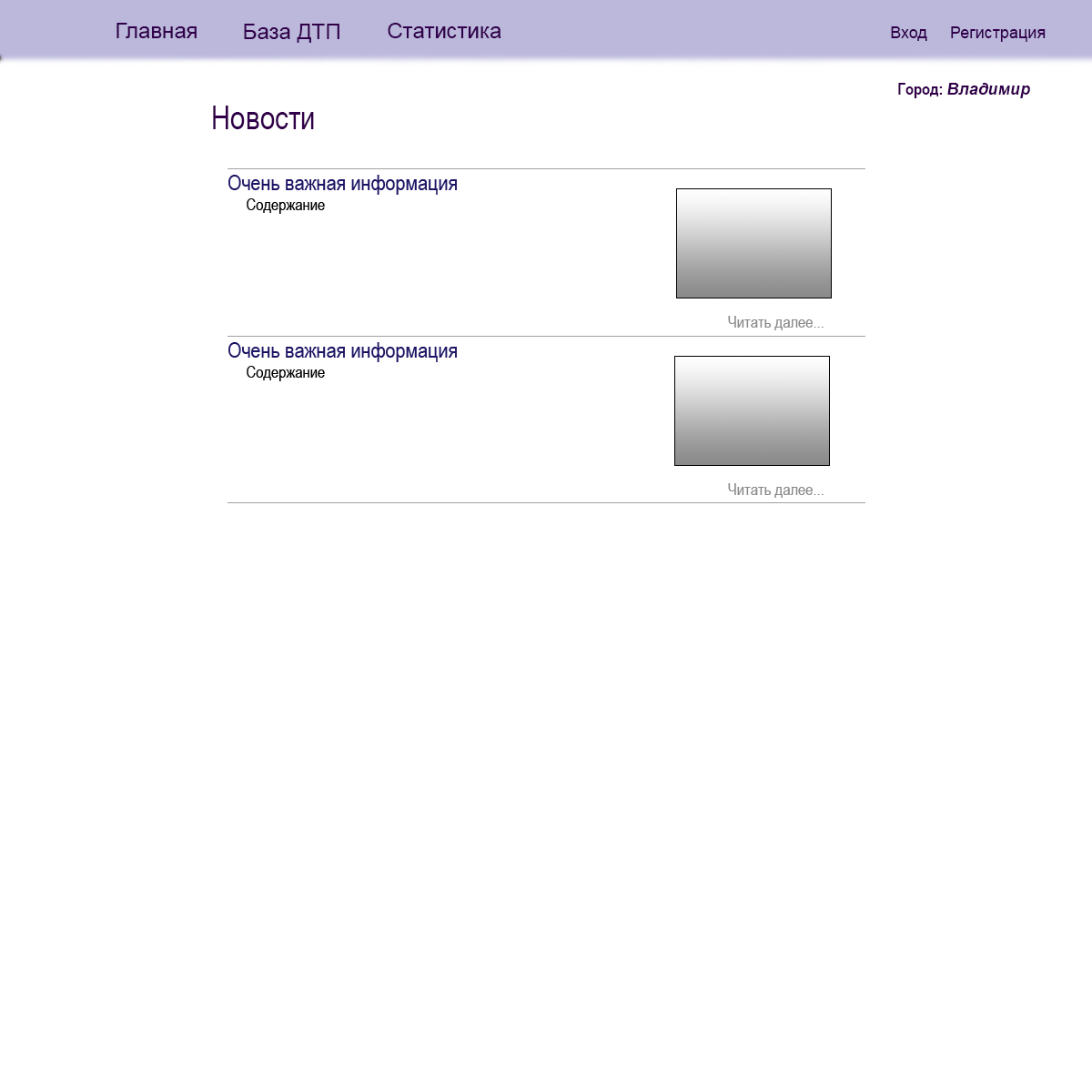


Рисунок 14. Страница новостей.